UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/883,508	06/19/2001	Jeffrey A. Bedell	53470.003042	8696

21967 7590 05/16/2005

HUNTON & WILLIAMS LLP
INTELLECTUAL PROPERTY DEPARTMENT
1900 K STREET, N.W.
SUITE 1200
WASHINGTON, DC 20006-1109

EXAMINER	
ZHEN, LI B	

ART UNIT	PAPER NUMBER
2194	

DATE MAILED: 05/16/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/883,508	BEDELL ET AL.	
	Examiner	Art Unit	
	Li B. Zhen	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 November 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1 – 18 are pending in the application.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

3. Claims 1 – 9 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

4. Currently amended claim 1 recites the new limitation "determining using a computer processor an appropriate manner of executing the selected function on the selected object" [lines 5 – 6] and "automatically causing the execution of the selected function on the selected object in the appropriate manner" [lines 11 – 12]. There does not appear to be a written description of the claimed limitation in the application as filed. Throughout the specification, applicant discloses that the user determines an appropriate manner of executing the selected function on the selected object. For example, specification discloses the system application determines whether to duplicate [selected function] the object [selected object] in the destination project or whether to take no action by prompting the user to input the desired result [emphasis added; p. 19,

lines 6 – 9]. Further down on the same page, applicant discloses the system application determines whether to keep the destination object as it is, replace the destination object with the source object, or duplicate the object by prompting the user [p. 19, lines 13 – 19]. Although the user is only prompted if the selected object already exists in the destination project, the scope of the claims is broad enough to include both situations [selected object does not exist in the destination project and the selected object already exists in the destination project]. When the selected object already exists in the destination object, an appropriate manner of executing the selected function on the selected object is determined by a user [not a computer processor] and the user causes execution of the selected function on the selected object [this step is not done automatically]. Therefore, the applicant fails to disclose "determining using a computer processor an appropriate manner of executing the selected function on the selected object" and "automatically causing the execution of the selected function on the selected object in the appropriate manner" in the specification as filed.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6. **Claims 1, 10 and 18 are rejected under 35 U.S.C. 102(b) as being anticipated by "MicroStrategy Administrator" Version 6.0 (hereinafter MicroStrategy) published on October 1999. This reference was cited in Information Disclosure Statement submitted on September 30, 2002.**

7. As to claim 1, MicroStrategy teaches a method for managing groups of objects [MicroStrategy Object Manager enables you to move objects easily by using familiar drag-and-drop and cut-and-paste actions; p. 27, 1st paragraph] for use in a reporting system project [EIS contains reporting modules; p. 94] comprising the steps of:

receiving a command [copying and moving objects] to perform a selected function on a selected object [MicroStrategy Object Manager allows you to open more than one project at a time, which makes it easy for you to transfer items between open project; p. 28, 1st paragraph];

automatically identifying dependent objects referred to by the selected object [finding object dependencies; p. 58 – 60];

determining using a computer processor an appropriate manner of executing the selected function [Click Copy on the dialog bar. Then highlight a selection and click Paste on the dialog bar to copy over the selection; p. 28, 4th paragraph];

determining using a computer processor appropriate functions to be performed on the dependent objects [sometimes when moving or copying an object with child objects, the child objects need to be copied as well; p. 28, 3rd paragraph];

automatically causing the appropriate functions to be performed on the dependent objects [MicroStrategy Object Manager will need to create one or more folders in the destination to hold the child objects; p. 28, 3rd paragraph]; and

automatically causing the execution of the selected function in the appropriate manner [MicroStrategy Object Manager checks to see whether an object of the same name already exists; if not, the copy operation is completed; p. 29, 1st – 3rd paragraphs].

8. As to claim 10, this is a system claim that corresponds to method claim 1; note the rejection to claim 1 above, which also meet this system claim.

9. As to claim 18, this is a product claim that corresponds to method claim 1; note the rejection to claim 1 above, which also meet this product claim.

10. **Claims 1, 3 – 5, 10 – 12 and 16 – 18 are rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Patent NO. 5,854,932 to Mariani.**

11. As to claim 1, Mariani teaches a computer implemented method for managing groups of objects [source code files 60 and header files] for use in a reporting system project [a development environment 52 which includes various tools 54 for creating, editing, and compiling source code files 60 and header files 62 for a user's programming project; col. 7, line 60 – col. 8, line 7] comprising the steps of:

receiving a command [modify] to perform a selected function on a selected object [With the editors 70, the user can directly modify or add C++ statements to the source code files 60 and header files 62; col. 8, lines 7 – 22];

automatically identifying using a computer processor dependent objects referred to by the selected object [for each of the object code files 82...the minimal rebuild system 100 generates dependency information...consisting of a set of classes...on which the respective object code file depends, and a set of dependency types...of the object code file on each class in the set; col. 12, lines 27 – 67];

determining using a computer processor an appropriate manner of executing the selected function [minimal rebuild system 100 determines when recompiling can be avoided by determining how the object code files 82 are dependent on the header files 62; col. 9, lines 1 – 15];

determining using a computer processor appropriate functions to be performed on the dependent objects [minimal rebuild system 100 selectively recompiles the source code files 60...so as to detect changes to all header files 62...that were changed since the last build of the project. The minimal rebuild system 100 then utilizes the detected changes to the header files 62 to determine which of the remaining source code files 60 to recompile and which recompiles can be avoided; col. 17, lines 18 – 30];

automatically causing the appropriate functions to be performed on the dependent objects [omit compiling selected source file and resave object file, step 174, Fig. 6B; compile selected source file into object file, step 176, Fig. 6B; col. 19, lines 33 – 67]; and

automatically causing the execution of the selected function on the selected object in the appropriate manner [minimal rebuild system 100 first recompiles the source code files that were changed since the user's project was last built; col. 17, lines 30 – 43].

12. As to claim 10, this is a system claim that corresponds to method claim 1; note the rejection to claim 1 above, which also meet this system claim.

13. As to claim 18, this is a product claim that corresponds to method claim 1; note the rejection to claim 1 above, which also meet this product claim.

14. As to claim 3, Mariani teaches the step of causing the appropriate functions to be performed on the dependent objects is performed prior to the step of causing the execution of the selected function in the appropriate manner [minimal rebuild system 100 reorders this list to place any source code files 60 that have changed since the user's project was last rebuilt first in the list; col. 17, lines 43 – 65].

15. As to claim 4, Mariani teaches the objects are grouped in projects and the selected function relates to manipulating objects within and between projects [For use in creating and editing the source code files 60 and header files 62 of the user's project, the development tools 54 of the environment 52 comprise one or more editors 70, and automated source code generators 72; col. 8, lines 7 – 22] and wherein within each

project each object has a unique identifier [search for a name in the scope of the class; col. 13, lines 25 – 50] and a version identifier [date and time of the last change made to the header files; col. 10, lines 8 – 45].

16. As to claim 5, Mariani teaches comparing dependent objects at a source and objects at a destination to determine whether an object at the destination exists in an identical form to each of the dependent objects at the source and whether an object at the destination exists in a modified form to each of the dependent objects at the source [minimal rebuild system 100 preferably determines which of the header files 62 were changed since the project was last built by comparing the date stamps of the header files to their last known date stamps; col. 18, lines 34 – 48] and determining, based on the step of comparing, which dependent object to copy from the source to the destination such that the selected object remains complete after execution of the selected function in the appropriate manner [the minimal rebuild system 100 selects the next source code file in the list that is not yet recompiled. At step 171, the minimal rebuild system 100 checks whether the object code file that was compiled in a previous build of the project from the selected source code file still exists; col. 19, lines 7 – 32].

17. As to claim 11, Mariani teaches the operational module interfaces [classes for interfacing with database management systems; col. 8, lines 23 – 43] with projects that reside in various environments [Class dependency information is similarly reduced for manually generated projects that use a class library; col. 12, lines 28 – 67].

18. As to claim 12, this is rejected for the same reasons as claim 3 above.

19. As to claims 16 and 17, these are rejected for the same reasons as claim 5 above.

Claim Rejections - 35 USC § 103

20. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

21. **Claims 2, 6 – 9 and 13 – 15 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mariani in view of U.S. Patent NO. 6,112,024 to Almond.**

22. As to claim 2, Mariani does not teach the selected object is contained in metadata of an on-line analytical processing system.

However, Almond teaches an object cycle versioning system [col. 2, lines 39 – 52] and an object contained in metadata [maps an object into a meta model which facilitates version control; col. 3, lines 1 – 54 and col. 6, lines 40 – 67] of an on-line analytical processing program [View reports--quickly view project activity status; col. 39, lines 15 – 39].

23. It would have been obvious to a person of ordinarily skilled in the art at the time of the invention to apply the teaching of storing an object in metadata of an on-line analytical processing system as taught by Almond to the invention of Mariani because this map objects to representations and allow operations supported by the system for versioning will execute correctly even if the objects are stored in a format other than a relational database, such as an object-oriented database, a file server, or other storage system [col. 3, lines 3 – 10 of Almond].

24. As to claim 6, Mariani as modified teaches the step of receiving is a step of receiving a command to copy a selected object from a source project to a destination project [Get--copy one or multiple objects to the user's local directory; col. 39, lines 16 – 39 of Almond].

25. As to claim 7, Mariani as modified teaches the unique identifier [search for a name in the scope of the class; col. 13, lines 25 – 50 of Mariani] and version identifier [date and time of the last change made to the header files; col. 10, lines 8 – 45 of Mariani] of objects in the source project are similar to the unique identifier and version identifier of objects in the destination project [minimal rebuild system 100 preferably determines which of the header files 62 were changed since the project was last built by comparing the date stamps of the header files to their last known date stamps; col. 18, lines 34 – 48 of Mariani].

26. As to claim 8, Mariani as modified teaches comparing unique identifiers [search for a name in the scope of the class; col. 13, lines 25 – 50 of Mariani] and version identifiers [date and time of the last change made to the header files; col. 10, lines 8 – 45 of Mariani], whether the selected object exists in the destination project in an identical form and whether the selected object exists in the destination project in a modified form [minimal rebuild system 100 preferably determines which of the header files 62 were changed since the project was last built by comparing the date stamps of the header files to their last known date stamps; col. 18, lines 34 – 48 of Mariani]; and

selecting whether to copy the selected object from the source project to the destination project [Get--copy one or multiple objects to the user's local directory; col. 39, lines 16 – 39 of Almond], to replace an object in the destination project with the selected object [compile selected source file into object file, step 176, Fig. 6B; col. 19, lines 33 – 67 of Mariani], and to keep an object in the destination project as is [omit compiling selected source file and resave object file, step 174, Fig. 6B; col. 19, lines 33 – 67 of Mariani].

27. As to claim 9, Mariani as modified teaches the step of selecting includes prompting the user for a selection [user can directly modify or add C++ statements to the source code files 60 and header files 62; col. 8, lines 7 – 23 of Mariani].

28. As to claim 13, Mariani as modified teaches the operation module interfaces

[RPC interface allows the system to surface an Object Cycle API...for development system clients; col. 2, lines 39 – 54 of Almond] with projects of an on-line analytical processing system [View reports--quickly view project activity status; col. 39, lines 15 – 39 of Almond].

29. As to claim 14, Mariani as modified teaches the operational module, upon receiving a user command to copy the selected object from a source project to a destination project [Get--copy one or multiple objects to the user's local directory; col. 39, lines 16 – 39 of Almond], determines, by comparing the unique identifiers [search for a name in the scope of the class; col. 13, lines 25 – 50 of Mariani] and the version identifiers [date and time of the last change made to the header files; col. 10, lines 8 – 45 of Mariani], whether the selected object exists in the destination project in an identical form and whether the selected object exists in the destination project in a modified form [minimal rebuild system 100 preferably determines which of the header files 62 were changed since the project was last built by comparing the date stamps of the header files to their last known date stamps; col. 18, lines 34 – 48 of Mariani].

30. As to claim 15, Mariani as modified teaches the operational module communicates with the user interface to select whether to copy the selected object from the source project to the destination project [Get--copy one or multiple objects to the user's local directory; col. 39, lines 16 – 39 of Almond], to replace an object in the destination object with the selected object [compile selected source file into object file,

step 176, Fig. 6B; col. 19, lines 33 – 67 of Mariani], and to keep an object in the destination project as is [omit compiling selected source file and resave object file, step 174, Fig. 6B; col. 19, lines 33 – 67 of Mariani].

Response to Arguments

31. Applicant's arguments filed November 8, 2004 have been fully considered but they are not persuasive.

In response to the Non-Final Office action dated July 6, 2004, applicant argues:

(1) MicroStrategy does not teach functions that are either accomplished without human intervention or by a computer processor [p. 7, lines 3 – 17];

(2) Mariani does not teach or suggest "determining using a computer processor a appropriate manner of executing the selected function" and the word "manner" refers to the way that something is done, not the fact of whether or not it is done [p. 9, lines 5 – 15];

(3) the Office Action has not shown a single, coherent embodiment of Mariani that teaches all the elements of the claims [p. 9, lines 16 – 25]; and

(4) Almond's reference to copying objects fails to teach the subject matter of claim 6 because this feature cannot be combined with Mariani in the manner prescribed by claim 6 [p. 10, lines 19 – 21]. Copying and recompiling are completed different operations that require completely different systems and methods; thus, the combination of Mariani and Almond does not teach or suggest claim 6 [p. 11, lines 7 – 12].

As to argument (1), see the **35 USC § 112** rejection above. As to human intervention, examiner notes that the specification of applicant's invention also discloses the need for human intervention. With regards to applicant's submission that MircoStrategy's copy functions are not determined by a computer processor, examiner notes that a user uses MicroStrategy's Object Manager to copy items between projects [p. 28, 1st paragraph]. The MicroStrategy Object Manager is an application executed by the computer processor. When the user interacts with the MicroStrategy Object Manager to input selected items for copying, the Object Manager program executed by the computer processor processes the user's selection to determine an appropriate manner of executing the user's selected function. The functions of the MicroStrategy Object Manager are accomplished with a computer processor.

In response to argument (2), examiner respectfully notes that the specification identifies an appropriate manner of executing the copy command as copying the object to the destination project [p. 18, line 21 – p. 19, line 1], duplicate the object [p. 19, lines 6 – 8], take no action [p. 19, lines 6 – 8], or replace the destination object [p. 19, lines 13 – 16]. Copying, duplicating and replacing an object require copying the source object. Therefore, the appropriate manner would include either copying the object or taking no action. Applicant's argument does not appear to be consistent with the specification. However, examiner notes that Mariani teaches "determining using a computer processor a appropriate manner of executing the selected function". Mariani teaches modifying source code files [command to perform a selected function] that includes determining whether to recompile source code files [appropriate manner; see the

rejection to claim above]. In other words, whether to recompile the source code files is an appropriate manner [separate or addition step] of performing source code modification [command to perform a selected function].

As to argument (3), examiner notes that "modifying code" in Mariani corresponds to a command to perform a selected function of the claims and "recompiling" in Mariani corresponds to an appropriate manner of executing the selected function of the claims [see also response to argument (2) above]. Additionally, examiner submits the "source code file" of Mariani corresponds to the selected object of the claims. The "object code file" in element (ii) refers to a dependency analysis method which identifies a set of classes [source code files] that the object code file depends [generates dependency information...consisting of a set of classes...on which the respective object code file depends; col. 12, lines 28 – 36 of Mariani].

In response to argument (4), examiner respectfully disagrees and submits that the combination of Mariani and Almond teaches claim 6. Both Mariani and Almond teaches a project management system [col. 3, line 65 – col. 4, line 15 of Mariani and col. 2, lines 39 – 52 of Almond] and it is obvious to back-up or copy a project before the project rebuilt to provide an archive of builds for the project.

Conclusion

32. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

33. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Li B. Zhen
Examiner
Art Unit 2194

lbz

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100